Efficient Mining of Top-k Breaker Emerging Subgraph Patterns from Graph Datasets

Min Gan **Honghua Dai**

School of Information Technology
Deakin University
Melbourne, Victoria 3125, Australia
Email: {min.gan.au, honghuadai}@gmail.com

Abstract

This paper introduces a new type of discriminative subgraph pattern called breaker emerging subgraph pattern by introducing three constraints and two new concepts: base and breaker. A breaker emerging subgraph pattern consists of three subpatterns: a constrained emerging subgraph pattern, a set of bases and a set of breakers. An efficient approach is proposed for the discovery of top-k breaker emerging subgraph patterns from graph datasets. Experimental results show that the approach is capable of efficiently discovering top-k breaker emerging subgraph patterns from given datasets, is more efficient than two previous methods for mining discriminative subgraph patterns. The discovered top-k breaker emerging subgraph patterns are more informative, more discriminative, more accurate and more compact than the minimal distinguishing subgraph patterns. The top-k breaker emerging patterns are more useful for substructure analysis, such as molecular fragment analysis.

Keywords: Breaker emerging subgraph patterns, discriminative patterns, graph mining.

1 Introduction

As an abstract data structure, graphs are suitable for representing any objects and their relationships. A graph is a set of vertices and edges, where a vertex represents an object, and an edge between two vertices represent that a relationship exists between the two vertices. In real world, there exist large amounts of data that can be represented as graphs, such as molecular structures, Web-link structures, biological networks, transport networks and social networks. Graph mining mainly studies how to discover knowledge from graph data. In recent years graph mining has become an active research field. Many graph mining methods have been proposed for discovering various patterns from graph data. No matter what methods are used and what patterns are discovered, many graph mining tasks need to conduct a key operation, graph comparison, which detects two kinds of information: graph similarity and graph dissimilarity. This paper focuses on graph dissimilarity.

Graph dissimilarity reflects the difference between two graphs or two classes of graphs. Conventional graph matching metrics such as graph edit distance (Sanfeliu et al. 1983), maximal common subgraphs (McGregor 1982) and subgraph isomorphism can be used to measure the dissimilarity (as well as the similarity) between two graphs. However, these metrics are not applicable to measuring the dissimilarity between two contrasting classes of graphs, which is a key issue in graph mining. From an application point of view, in graph mining, there exist many cases in which one needs to detect the dissimilarity between two contrasting classes of graphs. For example, in drug analysis, medical experts detect molecular differences between two classes of drug components (strong side effect vs. weak side effect) to explore the molecular mechanism of the side effect. In e-commerce website analysis one detects differences of Web access behaviors between two classes of visitors (purchaser vs. non-purchasers or males vs. females) to improve website organization or provide customized Web-link structures. From the point of view of data mining theory, the differential information between two contrasting classes of data is crucial for many mining tasks such as classification. In order to distinguish from the dissimilarity between two individual graphs, in this paper "graph class dissimilarity" is used to denote the dissimilarity between two classes of graphs.

Graph class dissimilarity is usually represented as discriminative subgraph patterns. Therefore, the first problem in detecting graph class dissimilarity is: which patterns are the best to be used for identifying graph class dissimilarity. The second problem is how to discover the patterns efficiently. Discriminative subgraph patterns can be classified into two categories: one is discriminative individual (connected) subgraph patterns, and the other is discriminative multiple subgraph patterns (a pattern consists of one or multiple connected subgraphs). The former is normally used for individual substructure analysis. The latter is usually more discriminative than the former and is used for effective classification. The two types of patterns are more complementary than competitive. In this paper, we focus on discriminative individual subgraph patterns.

Most existing discriminative patterns are only for simple types of data, such as transactional data and relational data, and few discriminative patterns for graph data have been proposed. As an important discriminative pattern, emerging pattern (EP) (Dong et al. 1999) has been proved to be of strong discriminating power for distinguishing between two classes of data, and has broad applications, such as construction of accurate classifiers (Ramamohanarao et al. 2006). In recent years, researchers have extended the discovery of emerging patterns from simple types of data to graph data, and proposed two kinds of patterns: contrast subgraph pattern (CSP) (Ting et al. 2006) and distinguishing subgraph pattern (DSP) (Zeng et al. 2008). Recently Fan et al. proposed a general discriminative pattern, discriminative and essential frequent pattern (DEFP) (Fan et al. 2008), which applies to various types of data including graph data.

However, we found that none of CSP and DSP include the most discriminative individual subgraph patterns exactly, and both of them have some drawbacks as analysed in the next section. The DEFP is essentially discriminative and has been applied to effective classification (Cheng et al. 2008), but as a kind of discriminative multiple subgraph pattern, it is not applicable to individual substructure analysis. Our study aims at introducing a more accurate and more informative discriminative individual subgraph pattern, and devising an efficient mining algorithm. In this paper, we introduce a new type of discriminative subgraph pattern called breaker emerging subgraph pattern (BESP), and devise an efficient algorithm to discover the top-k BESPs from graph datasets.

The rest of this paper is organised as follows. Related work is reviewed and analysed in Section 2. Motivations are illustrated in Section 3. Section 4 defines the breaker emerging subgraph pattern. Section 5 proposes an efficient algorithm for mining top-k BESPs. Experimental results are presented in Section 6. Conclusions and future work are included in Section 7.

2 Related Work

In this section, we provide a brief summary of the related work on emerging patterns, contrast subgraph patterns and distinguishing subgraph patterns.

2.1 Emerging Pattern

The emerging pattern (EP) was originally proposed by Dong et al. (1999). An EP is defined as an itemset X whose support increases significantly from one dataset D_N to another, D_P, where the increasing degree of the support is measured by growth rate, which is defined as

$$GR_{D_N \to D_P}(X) = \begin{cases} 0 & \text{if } supp_N(X) = 0 \\ & \text{and } supp_P(X) = 0 \\ \infty & \text{if } supp_N(X) = 0 \\ & \text{and } supp_P(X) \neq 0 \\ \frac{supp_P(X)}{supp_N(X)} & \text{otherwise} \end{cases} \tag{1}$$

where $supp_P(X)$ is the support of itemset X in D_P, which equals $count_P(X)/|D_P|$, $count_P(X)$ is the total number of transactions in D_P that contain X, and $|D_P|$ is the total number of transactions in D_P. Similarly $supp_N(X)$ represents the support of X in D_N, which equals the number of transactions in D_N that contains X over the total number of transactions in D_N, denoted by $|D_N|$ (Dong et al. 1999).

Given the minimal growth rate threshold Min_GR, EPs from D_N to D_P are itemsets whose growth rates are no less than Min_GR (Dong et al. 1999). The higher GR of an EP is, the more discriminative and more significant the pattern is. In this paper the growth rate is also adopted to evaluate the discriminating power of a pattern.

Although the EP was originally defined on items by Dong et al. (1999), it applies to any other types of data including graph data. When it is defined on graph data, the EP can be called emerging subgraph pattern (ESP).

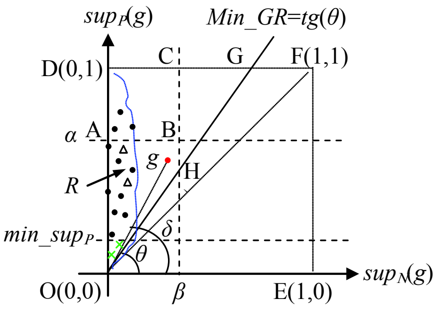


Figure 1: The support plane of emerging subgraph patterns

2.2 Contrast Subgraph Pattern

Contrast subgraph patterns (CSPs) (Ting et al. 2006) are defined as subgraphs[1] that appear in one class of graphs D_P, but never appear in another class of graphs D_N. A CSP is minimal if none of its strict subgraphs are CSPs. For CSPs, only the minimal CSPs (MCSPs) are discovered (Ting et al. 2006).

2.3 Distinguishing Subgraph Pattern

The distinguishing subgraph pattern (DSP) was proposed by Zeng et al. (2008). Given two graph datasets D_P, D_N and two support thresholds α, β ($\alpha, \beta \in [0,1], \alpha \gg \beta$, where $\gg$ means much greater than), a subgraph g is a DSP if $supp_P(g) \geq \alpha$ and $supp_N(g) \leq \beta$. The pattern g is a minimal DSP (MDSP) if no strict subgraphs of g are DSPs. Among all DSPs, only MDSPs are discovered (Zeng et al. 2008).

2.4 Analysis

To illustrate and analyse the above patterns, in Fig.1 we use a plane rectangular coordinate system (similar to the support plane (Dong et al. 1999)) to represent any ESP g and its growth rate ($GR(g) = tg(\delta)$). The closer to line OD a point g is, the higher $GR(g)$ is, i.e., the more discriminative the ESP g is. In Fig.1, ESPs are the points in the triangle OGD, CSPs are the points on the line OD, and DSPs are the points in the rectangle $ABCD$.

The CSPs are the most discriminative ESPs with infinite growth rates. However, some problems occur when they are applied to real datasets. First, the constraint is so strict that sometimes no or only a few such patterns exist in the datasets. Second, CSPs are so sensitive to noise that false patterns could be involved in the result and some real patterns could be missed when the patterns are corrupted by noise. For example, (1) if a noise ∘ (a vertex or an edge) appears at least one time in D_P and never appear in D_N, then ∘ will be found as a MCSP; (2) assumed that $g' = g \diamond e$ (g' is extended from g by adding an edge e) is a real MCSP, and g appears in D_N, if e is added to one of the matches of g in D_N by mistake, then g' will be missed. In addition, the CSP is a kind of discriminative multiple subgraph pattern since disconnected graphs are permitted. This disconnectness allowance blows up the search space (Ting et al. 2006) and makes it not applicable to the scenario of individual substructure analysis.

With the thresholds α and β, the DSP is not so sensitive to the noise with low frequencies[2]. However, to obtain significantly discriminative patterns, α is

[1] Both connected subgraphs and disconnected subgraphs are permitted

[2] The frequencies of the noise are assumed to be lower than the support threshold in this paper

usually needed to be specified a very high value and β a very low value. With this specification, discriminative patterns in the quadrangle $ABHO$ in Fig.1 will be missed. Another drawback of MDSP is that some more discriminative patterns could be missed as MDSPs are not necessarily the most discriminative. For example, if g is a MDSP, then all super-patterns of g will not be included in the result. Thus, more discriminative patterns (g's super-patterns with higher GR values) will be missed.

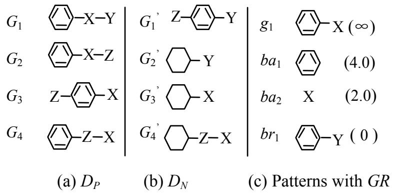
Another choice for discriminative subgraph patterns is a complete set of emerging subgraph patterns. However, it is not practicable as finding all ESPs is of high time-complexity, and in real applications, usually users are only interested in the k most discriminative patterns rather than all of them. In Fig. 1 the real top-k most discriminative patterns are the k black circles in region R (between line OD and the dotted curve) with green crosses (false patterns corrupted by noise) and white triangles (redundant patterns) filtered. However, as shown in Fig.1, both CSPs and DSPs only include part of the black circles. Additionally, as analysed above, the discovered MDSPs and MCSPs could be inaccurate with the risk of missing highly discriminative patterns and containing false patterns in the result. Moreover, redundant patterns are not considered and filtered in both CSP and DSP.

3 Motivations

As analysed above, none of the existing patterns, ESPs, MCSPs and MDSPs, include the top-k most discriminative subgraph patterns exactly, and no approaches have been proposed for mining top-k discriminative subgraph patterns. Therefore, it is necessary to introduce a more discriminative and more accurate pattern, and devise an efficient algorithm for the discovery of top-k such patterns.

Furthermore, we identify that none of the existing patterns include the information of patterns' structure changes and discriminating power changes. In substructure analysis, this change information is important. For example, a commonly used principle in chemistry and medicine domains is that structurally similar compounds are more likely to exhibit similar properties (Bender et al. 2004). The principle reflected by grow rates is that structurally similar subgraphs have comparative grow rates. An exception of the principle is that two structurally similar compounds exhibit different properties, i.e., the difference between their growth rates is very big. These two classes (normal and exceptional) of change information are interesting and significant for exploring a pattern's structure change and its impact on the property. In our new discriminative subgraph pattern, the two classes of change information are represented by two subpatterns called "base" and "breaker" respectively. The basic idea is illustrated by an example as follows.

Example 1 *Given two graph datasets D_P (Fig.2(a)) and D_N (Fig.2(b)) which consist of molecular structures of two contrasting classes of compounds respectively, assume that the compounds in D_P exhibit a positive property (e.g., toxicity) and the compounds in D_N exhibit the corresponding negative property (e.g., non-toxicity). The vertex labels X, Y and Z are abstract representations of concrete atoms, and the implicit vertex labels in the rings correspond to atom C (carbon). Given Min_GR = 2.0, the discovered top-1 ESP is g_1 in Fig.2(c). We examine the structure change and growth rate change of the patterns in Fig.2(c) (growth rates are in the brackets).*



(a) D_P (b) D_N (c) Patterns with GR

Figure 2: Graph datasets and subgraph patterns

The pattern g_1 indicates that compounds containing g_1 are likely to exhibit the positive property. We examine g_1's structure changes and growth rate changes. Considering ba_1 and ba_2 in Fig.2 (c), we notice that ba_1 and ba_2 are subgraphs of g_1, and they are two minimal ESPs, i.e., there exist no subgraphs of ba_1 and ba_2 that are ESPs. Intuitively ba_1 and ba_2 can be seen as two "bases" of g_1. Then we consider br_1. The pattern br_1 is structurally similar to g_1 since br_1 can be formed from g_1 by replacing the atom "X" with "Y". We notice that g_1's growth rate decreases sharply from ∞ to 0. The br_1 appears in D_N but never appears in D_P. This indicates that g_1 loses the positive property and exhibits strong negative property after being "broken" by replacing the atom "X" with "Y". The pair of br_1 and the operation can be seen as a "breaker" of g_1. For experts this information is not only useful for exploring the inner molecular mechanism of the property but also helpful for finding out ways to weaken or remove the property.

It is obvious that the ESPs with the "bases" and "breakers" are more informative. This type of ESP is called breaker ESP in this paper. This paper aims at defining the breaker ESP, and proposing an efficient approach to discover top-k breaker ESPs.

4 Breaker Emerging Subgraph Pattern

In this section, a new type of discriminative subgraph pattern, breaker ESP (BESP) is defined. After the definition of preliminary concepts, three constraints are introduced into ESPs; then the two subpatterns, base and breaker, are introduced; finally the BESP is defined.

4.1 Preliminary Concepts

The graphs considered in this paper are undirected labeled graphs.

Definition 1 (Undirected Labeled Graphs) *An undirected labeled graph G can be represented by a 5-tuple, $G = \{V, E, \Sigma_V, \Sigma_E, \lambda\}$, where V is a nonempty set of vertices, $E \subseteq V \times V$ is a set of undirected edges, Σ_V and Σ_E are the sets of vertex labels and edge labels respectively. The function λ defines the mappings from vertices to vertex labels, $V \rightarrow \Sigma_V$, and from edges to edge labels, $E \rightarrow \Sigma_E$.*

Definition 2 (Subgraphs) *G is a subgraph of G' (denoted by $G \subseteq G'$) iff $V \subseteq V'$ and $E \subseteq E' \cap (V \times V)$. $G \subset G'$ denotes G is a strict subgraph of G'.*

Definition 3 ((Sub)Graph Isomorphism) *Graph $G = \{V, E, \Sigma_V, \Sigma_E, \lambda\}$ is graph isomorphic to another graph $G' = \{V', E', \Sigma'_V, \Sigma'_E, \lambda'\}$ iff there exists a bijection $f : V \rightarrow V'$ such that for $\forall u \in V, f(u) \in V'$ and $\lambda(u) = \lambda'(f(u))$, and for $\forall e = (u, v) \in E$, $e' = (f(u), f(v)) \in E'$ and $\lambda(e) = \lambda'(e')$. Graph G is subgraph isomorphic to G' if there exists a subgraph G'' of G' such that G is graph isomorphic to G''.*

Definition 4 (Growth Rate of a Subgraph) *Given two graph datasets D_P and D_N, the growth rate of a subgraph g from D_N to D_P, $GR_{D_N \to D_P}(g)$, is defined as equation (1) (X is replaced by g).*

Definition 5 (Emerging Subgraph Patterns) *Given two graph datasets D_P, D_N and a threshold of growth rate, Min_GR, a set of emerging subgraph patterns (ESPs) from D_N to D_P is defined as:*

$$ESP_{D_N \to D_P} = \{g | GR_{D_N \to D_P}(g) \geq Min\_GR\} \quad (2)$$

It should be remarked that in the rest of the paper, the subscript $D_N \to D_P$ is omitted when it is apparent, i.e., $GR(g) = GR_{D_N \to D_P}(g)$, $Min\_GR = Min\_GR_{D_N \to D_P}$ and $ESP = ESP_{D_N \to D_P}$. Similarly, $GR_{D_P \to D_N}$ and $ESP_{D_P \to D_N}$ can be defined. For convenience, in the rest of the paper, the subscript "N" is used to denote "$D_P \to D_N$" instead, i.e., $GR_N(g) = GR_{D_P \to D_N}(g)$, $Min\_GR_N = Min\_GR_{D_P \to D_N}$ and $ESP_N = ESP_{D_P \to D_N}$.

Definition 6 ((Maximum) Common Subgraph) *A common subgraph of G_1 and G_2 is a graph G such that there exist subgraph isomorphism from G to G_1 and from G to G_2. We call G a maximum common subgraph of G_1 and G_2, $MCS(G_1, G_2)$, if there exists no other subgraph of G_1 and G_2 that has more vertices than G (Wang et al. 2005).*

4.2 Three Constraints on ESPs

As analysed in Section 2.4, the existing patterns, ESPs, MCSPs and MDSPs, have some drawbacks. The constraints α and β for MDSPs lead to the risk of missing highly discriminative patterns. Both ESPs and MCSPs have no constraints on support. This leads to two problems: one is huge searching complexity and the other is the inaccuracy due to the noise of low frequencies. Additionally, redundant patterns are not filtered in the three patterns. To overcome these drawbacks, we exert three constrains on ESPs.

The first constraint is the minimal support threshold. In our definition, any ESP g must be frequent, i.e., $supp_P(g) \geq min\_supp_P$, where min_supp_P is the threshold of $supp_P$. This constraint brings three advantages: (1) it ensures that the patterns are popular to some degree in the dataset; (2) it filters the noise with low frequencies, and thus filters some false patterns corrupted by the noise; (3) it greatly reduces the number of patterns that need to be generated, and thus reduces the computational complexity.

The second constraint is that any ESP g must be closed in D_P, that is to say there exist no proper supergraphs of g that have the same support of g. The first reason for exerting this constraint is that it can further reduce the number of patterns that need to be generated. The second reason is that it can filter some redundant patterns without losing significant ESPs. For example, if g is not closed, i.e., $\exists g'$ such that $g \subseteq g'$ and $supp_P(g) = supp_P(g')$, then $GR(g) \leq GR(g')$ since $sup_N(g) \geq sup_N(g')$. Therefore, for g', g is redundant. After adding the constraint, g will be pruned.

The third constraint is for pruning redundant patterns that have subgraphs or super-graphs with higher growth rate values. For a pair of ESPs g and g' having the relationship: $g \subset g'$ (or $g' \subset g$), (1) if $GR_P(g) > GR_P(g')$ then g' is pruned as a redundant pattern; (2) if $GR(g) = GR(g')$ and $supp_P(g) \neq supp_P(g')$ then the larger graph is pruned as a redundant pattern.

Given D_P, D_N, min_supp_P and Min_GR, the ESPs that satisfy the min_supp_P constraint is called frequent ESPs (FESPs), and the ESPs that satisfy the first two constraints above are called closed frequent ESPs (CFESPs), and the ESPs that satisfy the three constraints are called constrained ESPs (CESPs). The set of FESPs, CFESPs and CESPs are denoted by $FESP$, $CFESP$ and $CESP$ respectively.

4.3 Breaker Emerging Subgraph Pattern

As indicated in Example 1, besides each ESP itself, the bases and breakers of the pattern should be provided. A breaker ESP consists of three subpatterns: a CESP, a set of bases and a set of breakers.

The bases of a CESP g_i are defined as the minimal CFESPs in the subgraphs of g_i, which are formally defined below.

Definition 7 (The bases of a CESP) *Given D_P, D_N, min_supp_P and a set of CESPs, $CESP = \{g_i\}$, for $\forall g_i \in CESP$, the set of bases of g_i, Ba_i, is defined as*

$$Ba_i = \{ba | ba \in CFESP, ba \subset g_i, \textit{ and } \neg \exists s \in CFESP \textit{ such that } s \subset ba\} \quad (3)$$

As shown in Example 1, a breaker pattern, br, of a CESP g_i is structurally similar to g_i, but its grow rate decreases significantly. Two types of breakers are interesting. One is that br still appears frequently in D_P, but its growth rate is weaken to a value below Min_GR. The other is that br appears more frequently in D_N than in D_P, i.e., $GR_N(br) > 1$. The first type exhibits the same property as g_i to some extent, while the second type exhibits the opposite property. The first type is called weakening breaker, and the second type is called reverse beaker. To define the breakers, two metrics are needed to measure the structural similarity and the change degree of growth rate.

For real graph data from different applications, the standards for measuring the structural similarity could vary. Even in the same domain such as chemistry, dozens of similarity coefficients are available for measuring the structural similarity (Nikolova et al. 2004). In this paper, we just adopt a commonly used metric, the maximum common subgraph, to measure the structural similarity. The similarity degree between g_i and a candidate breaker pattern br is quantified by:

$$Similarity(g_i, br) = \frac{2|MCS(g_i, br)|}{|g_i| + |br|} \quad (4)$$

where, $|g_i|$ refers to the size of g_i. Two patterns are structurally similar if their $Similarity$ is no less than a user specified threshold $\delta \in (0, 1)$.

The graph size can be evaluated by edge number or vertex number. The $Similarity$ is denoted by $Similarity1$ ($Similarity2$) when vertex (edge) number is used.

For the first type of breaker, the change degree of the grow rate of a breaker candidate, br, of g_i, can be defined as

$$GR\_change(g_i, br) = \begin{cases} \infty & \text{if } GR(br) = 0 \\ \frac{GR(g_i)}{GR(br)} & \text{otherwise} \end{cases} \quad (5)$$

The change degree is significant if $GR\_change(g_i, br)$ is no less than a user specified threshold $\rho > 1$.

For the second type of breaker, GR_N represents the change degree, i.e., the degree that a breaker pattern exhibits the negative property.

The two types of breakers are formally defined as follows.

Definition 8 (Weakening Breaker) *Given D_P, D_N, $CESP = \{g_i\}$, $CFESP$, Min_GR, δ and ρ, for $\forall g_i \in CESP$, the set of weakening breakers of g_i, WBr_i, is defined as*

$$WBr_i = \{\langle br, \varphi\rangle | br \in CFESP, br = \varphi(g_i), \\ 1 \leq GR(br) < Min\_GR, Similarity(g_i, br) \geq \delta, \\ GR\_change(g_i, br) \geq \rho\} \quad (6)$$

A breaker of g_i consists of a breaker pattern *br* and a breaker operator φ, which is a set of operations that transforms g_i to *br*, and $br = \varphi(g_i)$ means that *br* can be formed by conducting the operator φ on g_i. The operations in φ are from 6 basic operations on graphs: AV (adding a vertex), AE (adding an edge), DV (deleting a vertex), DE (deleting an edge), MV (modifying a vertex label) and ME (modifying an edge label). Since the bases reflect the information of the patterns with GR no less than Min_GR, $GR(br)$ is constrained to be less than Min_GR.

Definition 9 (Reverse Breaker) *Given D_P, D_N, $CESP = \{g_i\}$, min_sup_N, Min_GR_N and δ for $\forall g_i \in CESP$, the set of reverse breakers of g_i, RBr_i, is defined as*

$$RBr_i = \{\langle br, \varphi\rangle | br = \varphi(g_i), Similarity(g_i, br) \geq \delta \\ sup_N(br) \geq min\_sup_N, GR_N(br) \geq Min\_GR_N\} \quad (7)$$

The threshold min_sup_N is used to ensure that *br* is popular to some extent in D_N and to filter the noise with low frequencies in D_N. The Min_GR_N is used to ensure that *br* loses the positive property and exhibits the negative property to some degree.

Based on the CESP and the definitions of base and breaker, breaker ESPs (BESPs) are defined as follows.

Definition 10 (Breaker ESPs) *Given D_P, D_N, min_sup_P, min_sup_N, Min_GR, Min_GR_N, δ and ρ, the set of BESPs from D_N to D_P is defined as*

$$BESP = \{\langle g_i, Ba_i, Br_i\rangle | g_i \in CESP\} \quad (8)$$

where, Ba_i is the set of bases of g_i and Br_i is the set of breakers of g_i ($Br_i = WBr_i \cup RBr_i$).

A breaker ESP is composed of three subpatterns: a constrained ESP, g_i, a set of bases Ba_i of g_i, and a set of breakers Br_i of g_i. It should be noted that Ba_i (Br_i) is an empty set when no bases (breakers) of g_i exist in the datasets. In implementation, the BESPs are sorted by the growth rate of g_i in descending order, and only the top-k BESPs are discovered, where k is a user-specified integer.

5 Mining Top-k Breaker Emerging Subgraph Patterns

An efficient algorithm for mining top-k BESPs, k-MBESP, is proposed in this section. The top-k BESPs are discovered in three main stages:

1. Find top-k constrained ESPs, $CESP_K = \{g_1, ..., g_k\}$;
2. Find the bases of each $g_i \in CESP_K$;
3. Find the breakers of each $g_i \in CESP_K$.

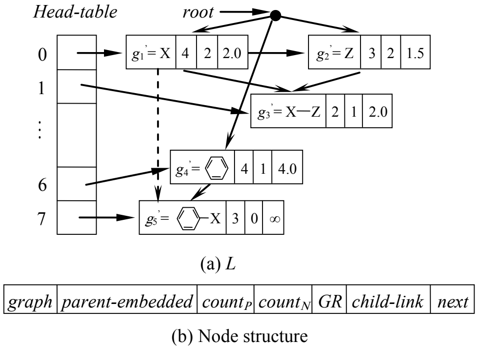
5.1 Finding Top-k Constrained ESPs

The top-k CESPs are found by 4 steps. First, the set of closed frequent subgraphs, CF, is discovered from D_P. Second, all closed frequent subgraphs are inserted into a layered graph L as shown in Fig.3. Third, D_N is scanned once to compute sup_N and GR of each subgraph in L. Finally, the top-k CESPs are detected and output from L. The procedure is described in Algorithm 1.

Algorithm 1 Top-k-CESP

Comments: find top-k CESPs.
Input: D_P, D_N, min_sup_P, Min_GR, k
Output: Top-k CESPs, $CESP_K$

```
1: $CESP_K \leftarrow \emptyset$;
2: Scan $D_P$ once to find frequent vertices $FV$;
3: for each vertex $v \in FV$ do
4:     CloseGraph($v$, $NULL$, $D_P$, $min\_sup_P$, $CF$);
5: for each graph ${G_i}'$ in $D_N$ do
6:     GR-Computation($L, {G_i}', |D_P|, |D_N|, Min\_GR$);
7: Traverse $L$ to single out $CESP_K$;
```



(a) L

(b) Node structure

Figure 3: The layered graph L and its node structure

In Line 4, the CloseGraph algorithm (Yan et al. 2003) is adopted to find CF first. In the implementation of CloseGraph, an additional subprocedure is addted to insert each found frequent closed subgraph into L as shown in Fig.3(a). In Line 6, GR-Computation computes current $count_N$ and GR of each node in L. The following example is used to illustrate the algorithm.

Example 2 *Given D_P and D_N as shown in Fig.2(a)(b), let min_sup_P=0.5, Min_GR=2.0, min_sup_N=0.25, Min_GR_N=2.0, δ = 0.8, ρ = 10 and k = 2. The k-MBESP algorithm is used to discover top-2 BESPs.*

In Example 2, firstly, $FV = \{C, X, Z\}$ is found; then CloseGraph generates $CF = \{{g_1}', {g_2}', ..., {g_5}'\}$, and inserts each ${g_i}' \in CF$ into L as shown in Fig.3(a). Each ${g_i}'$ and its related information are stored in a node of L. For space limitation, only 4 domains ($graph$, $count_P$, $count_N$ and GR) are shown explicitly. The node structure is shown in Fig.3(b). The L is organised in two dimensions. Horizontally, the graphs with the same edge number are organised in the same layer. Level numbers (edge numbers) are stored in the Head-table. Vertically, a child-link is set from a parent to a child (a node c is a child of node p if $p.graph \subset c.graph$). The root is chosen as its parent when a node has no parents. To avoid generating too many links, child-links are only set from the nearest parents to the children, e.g, a child-link is not set from ${g_1}'$ to ${g_5}'$.

The GR-Computation procedure is described in Algorithm 2. The basic idea is for each graph ${G_i}'$ in D_N, to search L from top to down to test whether the subgraph in each node is embedded in ${G_i}'$. To reduce the time complexity of subgraph-isomorphism test, two pruning strategies are introduced.

- **Pruning strategy 1** For node u in Level l, if $u.graph \not\subseteq {G_i}'$, then u's children are pruned

Algorithm 2 GR-Computation

```
Comments: compute $count_N$ and $GR$ of each node in $L$ after ${G_i}'$ is scanned.
Input: $L$, ${G_i}'$, $|D_P|$, $|D_N|$ and $Min\_GR$
Output: $L$ in which $count_N$ and $GR$ of each node have been computed after ${G_i}'$ is scanned
 1: for each node $u \in L$ do
 2:   $u.parent\text{-}embedded \leftarrow true$;
 3: for $l$=0 to $max\text{-}layer$ do
 4:   $u \leftarrow L.Head\text{-}table[l]$;
 5:   while $u \neq null$ do
 6:     if $u.GR \neq -1$ then
 7:       if $u.parent\text{-}embedded = true$ and
          $u.graph \subseteq {G_i}'$ then
 8:         $u.count_N \leftarrow u.count_N + 1$;
 9:         $u.GR \leftarrow \frac{u.count_P/|D_P|}{u.count_N/|D_N|}$;
10:         if $u.GR < Min\_GR$ then
11:           $u.GR \leftarrow -1$;
12:       else for each child $c$ of $u$ do
13:         $c.parent\text{-}embedded \leftarrow false$;
14:     if $i = |D_N|$ and $u.count_N = 0$ then
15:       $u.GR \leftarrow \infty$;
16:     $u \leftarrow u.next$;
```

Algorithm 3 Base-Detection

```
Comments: find the bases of each $g_i \in CESP_K$.
Input: $L$ and $CESP_K$
Output: $Ba_i (i = 1, 2, ..., k)$
 1: for each $Ba_i$ do $Ba_i \leftarrow \emptyset$;
 2: for each child $u$ of $L.root$
 3:   if $u.GR \neq -1$ then
 4:     for each $g_i \in CESP_K$
 5:       if $u.graph \subset g_i$ then
 6:         $Ba_i \leftarrow Ba_i \cup \{u.graph\}$;
```

(Note: this pruning does not mean really pruning the nodes from L but means that subgraph-isomorphism test need not be conducted between any graphs in the nodes and ${G_i}'$).

- **Pruning strategy 2** The subgraph-isomorphism test need not be done for node u if current $u.GR$ is less than Min_GR.

Pruning strategy 1 is implemented by Line 7, 12 and 13 in Algorithm 2. In Line 7, the second condition is tested only if the first condition is true. Pruning strategy 2 is implemented by Line 6, 10 and 11. A special value of -1 is used to indicate that current $u.GR$ is less than Min_GR. In Example 2, $count_N$ and GR of each node in L are computed by GR-Computation and their values are shown in Fig.3(a).

The last step of the Top-k-CESP procedure is, based on GR and the third constraint, traversing L and identifying top-2 CESPs, $CESP_2 = \{g_1 = {g_5}' : \infty,\ g_2 = {g_3}' : 2.0\}$.

5.2 Finding the Bases

The bases of each g_i in $CESP_K$ can be found easily from L since they are kept in L. Note that if there is a child-link from the root to node u, then $u.graph$ is a minimal CFESP. Therefore, based on Definition 7, if $u.graph \subset g_i$, then $u.graph$ is a base of g_i. The procedure is described in Algorithm 3. In Example 2, for g_1, $Ba_1 = \{g_1', g_4'\}$, and for g_2, $Ba_2 = \{g_1'\}$.

Algorithm 4 WBreaker-Identification

```
Comments: find the weakening breakers of each $g_i \in CESP_K$ and reverse breakers in $CFESP$
Input: $|D_P|$, $|D_N|$, $CESP_K$, $L$, $\delta$, $\rho$, $min\_sup_N$, $Min\_GR_N$ and $k$
Output: $WBr_i$, and $RBr_i$ if there exist reverse breaker patterns in $CFESP$ (i=1,2,...,k).
 1: for $i$=1 to $k$
 2:   Calculate $E_{min}(g_i)$ and $E_{max}(g_i)$;
 3:   Locate node $u$ in $L$ s.t. $u.graph = g_i$;
 4:   for each node $p$ from Layer $E_{min}(g_i)$ to
      Layer $E_{max}(g_i)$
 5:     if $p.GR = -1$ and ((there exists a path
        from $p$ to $u$ or from $u$ to $p$) or ($p$ and $u$
        have a common antecedent node from
        Layer $E_{min}(g_i)$ to Layer $|g_i| - 1$)) then
 6:       $brC_i \leftarrow brC_i \cup \{p.graph\}$;
 7: Scan $D_N$ to compute $count_N$ and $GR$ of each
    $br \in brC$ ($brC = \bigcup_{i=1}^{k} brC_i$);
 8: for each $g_i \in CESP_K$
 9:   for each $br \in brC_i$
10:     if $GR(br) \geq 1$ and $GR\_change(g_i, br) \geq \rho$
        then
11:       $WBr_i \leftarrow WBr_i \cup \{\langle br, \varphi \rangle\}$;
12:     else if $GR(br) < 1$, $1/GR(br) \geq$
        $Min\_GR_N$, $sup_N(br) \geq min\_sup_N$ then
13:       $RBr_i \leftarrow RBr_i \cup \{\langle br, \varphi \rangle\}$;
```

Algorithm 5 RBreaker-Identification

```
Comments: find the reverse breakers of each $g_i \in CESP_K$.
Input: $|D_P|$, $|D_N|$, $CESP_K$, $min\_sup_N$, $Min\_GR_N$, $\delta$, $k$
Output: $RBr_i (i = 1, 2, ..., k)$
 1: Find $FESP_N$ using Algorithm 1;
 2: for $i$=1 to $k$
 3:   for each $br \in FESP_N$
 4:     if $Similarity1(g_i, br) \geq \delta$ then
 5:       $RBr_i \leftarrow RBr_i \cup \{\langle br, \varphi \rangle\}$;
```

5.3 Finding the Breakers

Two breaker identification procedures are devised for the discovery of two types of breakers respectively.

5.3.1 Finding the Weakening Breakers

The procedure is described in Algorithm 4. In this procedure, edge number is used to evaluate graph size. First, obtain the minimum (maximum) edge number, $E_{min}(g_i)$ ($E_{max}(g_i)$) of candidate breaker patterns of each $g_i \in CESP_K$. Given δ and $|g_i|$, $E_{min}(g_i)$ and $E_{max}(g_i)$ can be derived easily from Equation (4) according to $|MCS(g_i, br)| \leq min\{|g_i|, |br|\}$. Second, traverse L from Layer $E_{min}(g_i)$ to $E_{max}(g_i)$ to detect candidate breaker patterns, brC_i, of g_i (Line 4, 5 and 6). Line 5 identifies the candidates that satisfy thresholds Min_GR and δ in Equation (6). Third, scan D_N to calculate $count_N$ and GR of each candidate. Finally check if the candidates satisfy the threshold ρ (Line 10). If there exists such br that satisfies the constrains of reverse breakers, then br is inserted into the corresponding reverse breaker set (Line 12 and 13).

5.3.2 Finding the Reverse Breakers

The procedure is described in Algorithm 5. In Line 1, the constraints *closed* and k are not needed in Algorithm 1 for finding frequent ESPs from D_P to D_N,

$FESP_N$. The key of the computation of $Similarity1$ (Equation (4)) in Line 4 is to identify the maximum common subgraph of g_i and br, $MCS(g_i, br)$. An existing efficient algorithm (Wang et al. 2005) is implemented to detect the minimal common subgraphs. The $MCS(g_i, br)$ is discovered by five major steps: (1) produce the matching pairs of two input graphs; (2) sort the order of matching pairs; (3) build a common subgraph path through selecting matching pairs; (4) determine the size of the corresponding common subgraph by the path; (5) continue finding paths until all paths have been considered. In $FESP_N$ usually almost no or only a small number of candidates are structurally similar to g_i. In order to accelerate the procedure, two pruning strategies are introduced to prune the search space.

- **MCS-pruning 1** If $\frac{2min\{|g_i|,|br|\}}{|g_i|+|br|} < \delta$ then $Similarity1(g_i, br) < \delta$ because $|MCS(g_i, br)| \leq min\{|g_i|, |br|\}$. In this case, $MCS(g_i, br)$ need not be identified.
- **MCS-pruning 2** In Step (1) of the MCS detection procedure, if the number of matching pairs, NMP, does not satisfy $\frac{2NMP}{|g_i|+|br|} \geq \delta$, then retreat from the procedure. Also in Step (3) only consider the pathes whose NMP satisfy $\frac{2NMP}{|g_i|+|br|} \geq \delta$.

In Example 2, the discovered reverse breakers are $Br_1 = \{\langle br_1, MV(v_7, Y)\rangle\}$, where br_1 is from Fig.2(c), and $MV(v_7, Y)$ means the label of v_7 (the vertex with label "X" in br_1) is modified to Y.

6 Experimental Results and Analysis

To evaluate the k-MBESP algorithm, experiments were conducted on both real and synthetic datasets. All experiments were done on a 2.2GHz Intel Core PC, with 2 GB main memory, running Windows XP. For comparison, we also implemented the algorithm for mining MCSPs (Ting et al. 2006) and the algorithm for mining MDSPs (Zeng et al. 2008), which are denoted by MCSP-Miner and MDSP-Miner respectively. All algorithms were implemented in Java.

6.1 Real Dataset

The real dataset that we use is the AIDS antiviral screen chemical compound dataset obtained from the website[3]. The dataset contains 42687 compounds, among which, 377 are confirmed active (CA), 1911 are confirmed moderately active (CM) and 40389 are confirmed inactive (CI). In the experiments, we only focus on CA and CI compounds. CA compounds are stored in D_P and CI compounds are stored in D_N. The k-MBESP algorithm is used to find the top-k BESPs from D_N to D_P. We determine an appropriate specification for the parameters: min_supp_P=14%, min_sup_N = 14%, k = 10, Min_GR=25.0, Min_GR_N=3.0, δ = 0.65, ρ = 8.0. With this specification, the algorithm finds top-10 BESPs within 2835 seconds. The top-1,2 and 7 BESPs are shown in Fig.4. The value in every bracket is the GR of a pattern. Several bases are found for each g_i, among which only a couple of them are shown in Fig.4. For g_7, a weakening breaker pattern, wbr_{7-1}, is discovered. We see that the growth rate of g_7 decreases from 88.76 to 8.82 ($GR\_change = 10.06$) when three bonds attached to atom S are broken as shown by the three dashed line in Fig.4. This information is heuristic and important for domain experts to discover the factors that could weaken the activity of the compounds. It should be noted that breaker patterns are not necessarily underlying in the datasets. No breaker patterns can be found if no breaker patterns exist in the datasets to be mined. For example, no reverse breakers are found in the AIDS dataset when the parameters are specified as above.

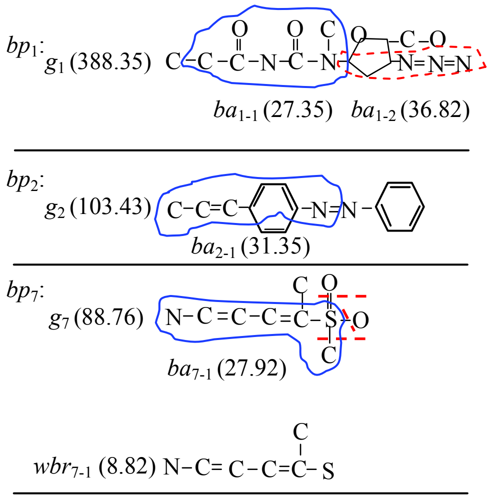


Figure 4: top-k BESPs discovered from the real dataset

In this dataset, MDSP-Miner (α = 14%, α/β = Min_GR = 25) only finds the minimum ESPs, and misses some more discriminative patterns. For example, some bases of g_1, g_2 are found as MDSPs and g_1, g_2 are excluded. However, MCSP-Miner is not able to finish the mining process within an acceptable time.

6.2 Synthetic Datasets

In order to evaluate the performance of the algorithm, we generated a series of synthetic datasets by a synthetic graph generator (Kuramochi et al. 2001) with fixed parameters I5T20L200V6E4 and varying D, where I denotes the average size of frequent patterns (in terms of edge number), T denotes the average size of graph transactions, L denotes the number of potentially frequent subgraphs, V denotes the number of distinct vertex labels, E denotes the number of distinct edge labels, and D denotes dataset size (the number of graphs in the dataset).

6.2.1 Performance Study

To compare the time efficiency, the three miners are performed on a series of datasets with the size varying from 20 to 100k. The parameters for k-MBESP are specified as: $Min\_GR = 25.0$, $min\_supp_P = 5\%$, δ (at most 2 vertices or edges are different), $\rho = 30.0$, Min_GR_N = 10.0, min_sup_N = 5% and k = 10. For MDSP-Miner, the parameter α = 5%, and the value of α/β is fixed at 25.0, which is as same as Min_GR. However, when the dataset size is 20 or 100, Min_GR = 5.0, min_supp_P = 10%, α = 30%, and β = 6%. Figure 5(a) shows a performance comparison of the three miners on datasets of 20 to 10000 graphs. As shown in Fig.5(a), k-MBESP is more efficient than two previous miners. When the dataset size is over 10k, both MCSP-Miner and MDSP-Miner are not able to finish the mining process in 3 hours. In contrast, k-MBESP can finish within 1000 seconds even on large datasets of up to 100k graphs. Fig.5(b) shows the runtime when D=5k and min_supp_P for k-MBESP (α for MDSP-Miner) varies from 1% to

[3] http://dtp.nci.nih.gov/docs/aids/aids_data.html

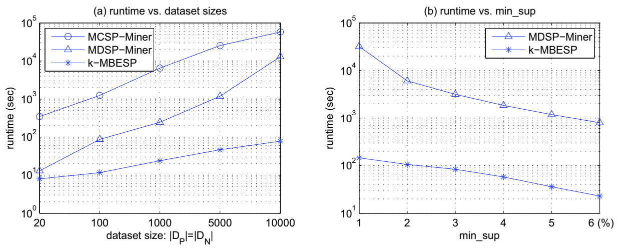


Figure 5: A performance comparison of the three miners

6%. We see that k-MBESP has better scalability on *min_sup* than MDSP-Miner.

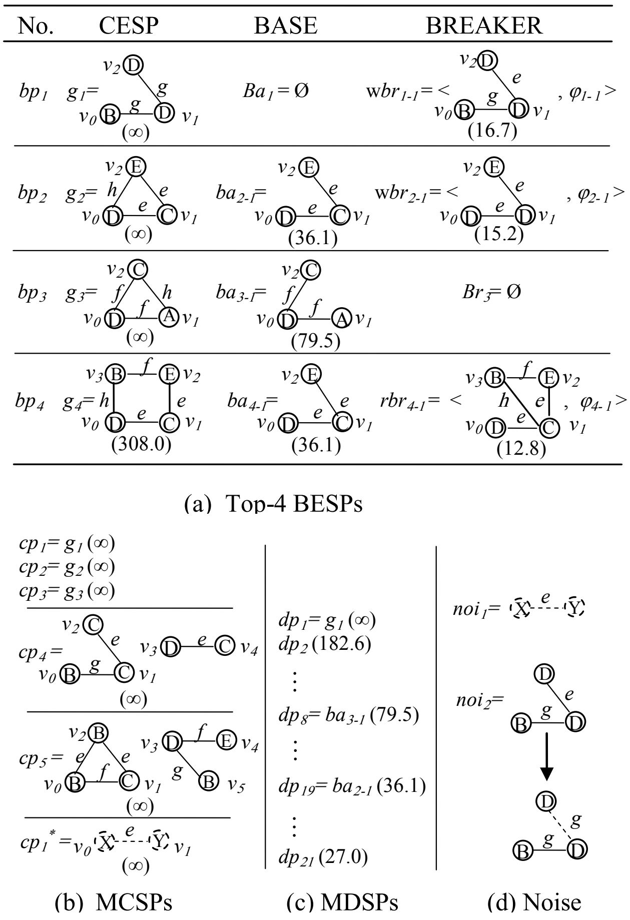
The high efficiency and good scalability of our algorithm benefit from the constraints, the compact data structure and the pruning strategies that we introduced. Firstly, the *min_sup* and *closed* constraints greatly reduce the search space of subgraph candidates. Secondly, the high compact layered graph contributes to the high efficiency. All candidates are stored in the layered graph which can be loaded into main memory prior to the computation of GR. Thus only one scan of the datasets is needed to compute GR for all candidates. In contrast, a large number of scans are required in the other two miners. In MDSP-Miner, one scan of the dataset is needed for each MDSP candidate, therefore the minimal number of scans is the number of MDSP candidates. In MCSP-Miner, for each graph in D_P, one scan of D_N is required for discovering the maximal common edge sets (Ting et al. 2006). consequently, for a dataset of 5k graphs, at least 5k scans are needed for MCSP-Miner. Thirdly, the pruning strategies further reduce the time complexity.

6.2.2 A Comparison of Discovered Patterns

We also compare the patterns discovered by the three miners to evaluate their informativeness, accuracy and discriminating power. Figure 6(a)(b)(c) show the patterns discovered by the three miners from datasets D5I5T20L200V6E4, which are denoted by bp_i, cp_i and dp_i respectively. In Fig.6(a) ba_{i-j} denotes the j^{th} base of g_i. The patterns in Fig.6(a)(c) are sorted according to GR values of CESPs and MDSPs respectively in descending order. As shown in Fig.6(a), for g_1 and g_2, two weakening breakers, wbr_{1-1} and wbr_{2-1}, are discovered, and both GR_change values are ∞. For g_4, a reverse breaker, rbr_{4-1}, is discovered and $GR_N = 12.8$. This indicates that g_4 loses the strong ($GR = \infty$) positive property and exhibits the negative property to some extent ($GR_N = 12.8$).

It is obvious that the top-k BESPs are more informative with the information of CESPs and their bases and breakers. Comparing bp_1 to cp_1 and dp_1 in Fig.6, we see that the CESP g_1 in bp_1 is just cp_1 and dp_1. The information of MCSPs (except disconnected subgraphs) and MDSPs are included in BESPs.

To test the accuracy of the miners, we introduce two noises in Fig.6(d) into the datasets: (1) a noi_1 is added into D_P, (2) a noi_2 is introduced into D_N by modifying a edge label of a subgraph in D_N from e to f. The result of K-MBESP is not affected by noi_1, and for bp_1, g_1's growth rate is changed to $count_P(g_1)$=1364. However, the noi_1 is found as a MCSP, cp_1*, by MCSP-Miner since $GR(noi_1) = 1/0 = \infty$, and when noi_2 appears, cp_1, i.e., g_1, can not be found by MCSP-Miner since $GR(cp_1) \neq \infty$. The result of MDSP-Miner is affected as same as that of k-MBESP by noi_1 and noi_2. In addition, MDSP-Miner could miss important patterns in the following two cases. First, as shown in Fig.6(c), g_2 is among the most discriminative patterns with infinite GR, but it is missed by MDSP-Miner since it is replaced by dp_{19}. It is clear that g_2 is more discriminative than dp_{19}, but g_2 is replaced by dp_{19} as $GR(dp_{19}) = 36.1 > 25.0$ and $dp_{19} \subset g_2$. Similarly, g_3 and g_4 are missed. Second, as shown in Fig.5(b), MDSP-Miner can not finish the mining process in an acceptable time when min_sup_P (i.e., α) is specified a very low value. Therefore, sometimes those MCSPs with low sup_P could not be found by MDSP-Miner. In contrast, K-MBESP can accept a relatively lower min_sup_P value. In addition, the constraints in BESPs filters some redundant patterns. Compared with the other two miners, k-MBESP is more accurate as it filters redundant patterns and false patterns corrupted by noise with low frequencies and does not miss more discriminative patterns.



(a) Top-4 BESPs

(b) MCSPs (c) MDSPs (d) Noise

Figure 6: A comparison of patterns discovered by three miners

As for discriminating power, the top-k BESPs are the top-k most discriminative patterns in terms of grow rate. In contrast, MDSPs are not necessarily the most discriminative, and some more discriminative patterns could be missed as examined above.

As for mining power, k-MBESP is more powerful than the other two miners. Figure 6(a) shows that k-MBESP is capable of discovering top-k BESPs. The other two miners can not discover them. One exception is that disconnected subgraph patterns such as cp_4 and cp_5 in Fig.6(b) are not considered in our miner since we only aim at detecting individual subgraph patterns of high discriminating power.

Another advantage is that, with relatively small number of patterns, top-k BESPs are more convenient for domain experts to select, examine and analyse. Furthermore, the change information of pattern structure and growth rate values kept in the bases and breakers provide heuristic information for the experts and help them discover important knowledge. In contrast, a relatively large number of MDSPs are not convenient for manual examination and analysis, and no change information is contained in both MCSPs and MDSPs.

In summary,the discovered top-k breaker emerging patterns are more informative, more discriminative and more accurate than the MCSPs and MDSPs extracted from the same datasets.

7 Conclusions and Future Work

In this paper, we introduced a new type of discriminative subgraph pattern, breaker emerging subgraph pattern, which consists of three important subpatterns: (1) the top-k CESPs that reflect the top-k most significant individual structural differences between two classes of graphs, (2) the bases that indicate structural bases of the discriminative patterns, and (3) the breakers that indicate triggers to weaken the growth rates of the patterns. We also proposed an efficient miner, k-MBESP, for the discovery of top-k BESPs. The experimental results show that the miner is capable of finding the top-k BESPs efficiently, more efficient, more powerful and more accurate than two previous miners. Compared with the complete sets of MSCPs and MDSPs discovered by previous miners, the top-k BESPs extracted by our algorithm have at least the following 4 advantages: (1) more informative (2) more discriminative in terms of growth rate, (3) more accurate, (4) more convenient and more useful for experts' further examination and analysis.

The BESP extends the application of discriminative subgraph patterns. It can be applied to: (1) detecting the difference between two contrasting classes of graphs, (2) exploring the inner structural mechanism of the property of a class of graphs, and (3) helping domain experts to discover ways to activate (strengthen) the desired properties, such as the activity to AIDS, and to break (weaken) the undesired properties, such as the toxicity.

Some future work needs to be done. First, more effective noise filtering strategies should be introduced to enhance the robustness of the top-k-BESP miner on data with various noise. Second, study the applications of the BESP. For example, use BESPs to detect differences between two contrasting classes of compounds or drugs (eg. high curative effects vs. low curative effects, high toxicity vs. low toxicity), to explore the molecular mechanism and to discover ways to design drugs of high curative effects and low toxicity. Based on the differences of website access behaviors between the males and the females represented by BESPs, modify the organization of a website to obtain a male-style website and a female-style one.

References

Agrawal, R., Imielinski, T. & Swami, A. (1993), Mining association rules between sets of items in large databases, *in* 'ACM SIGMOD International Conference on Management of Data', Vol. 22, ACM Press, Washington DC, USA, pp. 207–216.

Bender A. & Glen R. C., (2004), Molecular similarity: a key technique in molecular informatics, *Org. Biomol. Chem.* **2**, 3204–3218.

Cheng D., Yan Xi., Han J. & Yu P. S. (2008), Direct Discriminative Pattern Mining for Effective Classification, *in* 'International Conference on Data Engineering', pp. 169–178.

Dong, G. & Li, J. (1999), Efficient mining of emerging patterns: discovering trends and differences, *in* 'ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', Vol. 22, ACM Press, Washington DC, USA, pp. 207–216.

Fan W., Zhang K., Cheng H., Gao J., Yan X., Han J., Yu P. S. & Verscheure O. (2008), Direct mining of discriminative and essential frequent patterns via model-based search tree, *in* 'ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', ACM Press, Washington DC, USA, pp. 230–238.

Kuramochi M. & Karypis G. (2001), Frequent subgraph discovery, *in* 'IEEE International Conference on Data Mining', IEEE Computer Society, California, USA, pp. 313–320.

McGregor J. J. (1982), Backtrack search algorithms and the maximal common subgraph problem, *Software Practice and Experience* **12**, 23–24.

Nijssen S. & Kok J. N. (2004), A quickstart in frequent structure mining can make a difference, *in* 'ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', ACM Press, Washington DC, USA, pp. 647–652.

Nikolova N. & Jaworska J., (2004), Approaches to measure chemical similarity - A review, *QSAR Comb. Sci.* **22**, 1006–1026.

Ramamohanarao K., Bailey K. & Fan H. (2006), Efficient mining of contrast patterns and their application to classification, *in* 'International Conference on Intelligent Sensing and Information Processing', IEEE Computer Society, Washington DC, USA, pp. 39–47.

Sanfeliu A. & Fu K. S. (1983), A distance measure between attributed relational graphs for pattern recognition, *IEEE Trans. Systems, Man and Cybernetics* **13**, 353–362.

Ting R. M. H. & Bailey J., (2006), Mining minimal contrast subgraph patterns, *in* 'SIAM International Conference on Data Mining', Society for Industrial and Applied Mathematics, Philadelphia, USA pp. 639–643.

Wang y. & Maple C., (2005), A novel efficient algorithm for determining maximum common subgraphs, *in* 'The International Conference on Information Visualisation', IEEE Computer Society, Washington DC, USA, pp. 657–663.

Yan X. & Han J., (2003), CloseGraph: mining closed frequent graph patterns, *in* 'ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', ACM Press, Washington DC, USA, pp. 286–295.

Zeng Z., Wang J. & Zhou L., (2008), Efficient mining of minimal distinguishing subgraph patterns from graph databases, *in* 'The Pacific-Asia Conference on Knowledge Discovery and Data Mining', Springer-Verlag, Berlin Heidelberg, pp. 1062–1068.